Chapter 2.4
Automatic Partitioning of Large Scale Simulation in Grid Computing for Run Time Reduction

Nurcin Celik
The University of Arizona, USA

Omid Kazemi
The University of Arizona, USA

Esfandyar Mazhari
The University of Arizona, USA

Parag Sarfare
NetApp Inc., USA

John Canby
Schlumberger, Indonesia

Majed S. Al-Otaibi
The University of Arizona, USA

Young-Jun Son
The University of Arizona, USA

ABSTRACT

Simulating large-scale systems usually entails exhaustive computational powers and lengthy execution times. The goal of this research is to reduce execution time of large-scale simulations without sacrificing their accuracy by partitioning a monolithic model into multiple pieces automatically and executing them in a distributed computing environment. While this partitioning allows us to distribute required computational power to multiple computers, it creates a new challenge of synchronizing the partitioned models. In this article, a partitioning methodology based on a modified Prim's algorithm is proposed to minimize the overall simulation execution time considering 1) internal computation in each of the partitioned models and 2) time synchronization between them. In addition, the authors seek to find the most advantageous number of partitioned models from the monolithic model by evaluating the tradeoff between reduced computations vs. increased time synchronization requirements. In this article, epoch-based synchronization is employed to synchronize logical times of the partitioned simulations, where an appropriate time interval is determined based on the off-line simulation analyses. A computational grid framework is employed for execution of the simulations partitioned by the proposed methodology. The experimental results reveal that the proposed approach reduces simulation execution time significantly while maintaining the accuracy as compared with the monolithic simulation execution approach.

DOI: 10.4018/978-1-4666-0879-5.ch2.4

INTRODUCTION

Modern society depends upon many interacting large-scale dynamic systems, such as manufacturing, supply chain, defense, transportation, homeland security, and healthcare (Wysk et al., 2004). Although some of their constituent systems are amenable to mathematical and analytical analyses, the ensembles are not, because of scale, randomness, and complex dynamic response. Simulation is a primary modeling tool for analyzing and predicting the behavior of such large systems to support their design as well as operations. Among various simulation techniques, discrete-event simulation has become more widely used because it can take randomness into account, address aggregate as well as very detailed models, and enable a reusable experimentation set-up for analysis of such complex systems which is not possible or economical to be held in a real setting. While employing discrete-event simulations during the design stage of complex systems has become a common practice, high execution time and computational power usage become challenges when they are used to support short-term decisions (e.g. operations or maintenance scheduling).

An obvious, alternative approach is to execute a gigantic simulation in multiple computers in a distributed computing environment. However, it creates three new challenges. First, in order to execute a large scale simulation in a distributed manner, the component simulation models must be built in a divided manner during the development stage; or, if a large scale simulation already exists, it should be partitioned into smaller pieces (focus of this research). Here, it is quite challenging to devise a method to partition a monolithic model into multiple pieces to reduce its execution time without sacrificing their accuracy (Vardanega & Maziero, 2001; Kim et al., 1998). Second, simulation clocks of those partitions should be synchronized during the execution. Synchronization of different partitions has always been a challenging task and extensive research works have been performed in this area. The conservative synchronization (one of the major approaches) requires the partitioned models to ask for synchronization request every time an event occurs. Therefore, no information is lost when this type of synchronization is used. However, conservative synchronization approaches can introduce excessive overhead in execution, and result in little parallelism, which can eliminate the speedup promised by distributed simulation (Xu, 2006). Optimistic synchronization (another major approach) on the other hand suffers from a possibility of roll-back, which can cause computationally intense operations (Jefferson, 1985; Fujimoto, 2000). Third, as communications among the partitions during the execution may have a high impact on the performance of a distributed simulation, developing a reliable and efficient computational framework is another challenge (David et al., 1992). It is noted that as the number of partitions increases, the frequency of communications required by the partitions increases as well. Therefore, while it is beneficial to increase the number of partitions in terms of reduction of execution requirement for each individual simulation, it is at the same time disadvantageous in term of time synchronization and communication requirements. Therefore, an appropriate number of partitions must be determined considering both requirements.

In this research, we first assess and analyze the compromise between reduced computations and increased communication requirements in detail in partitioning of large scale simulations. Based on the analysis, we then propose a methodology to enable a near optimal partitioning of large-scale simulations into multiple pieces. Our methodology is novel for several reasons. First, while most of the previous studies have focused on simulation partitioning using a hierarchical topology of the existing models, we propose a new approach based on an efficient, polynomial-time heuristic algorithm (using a combination of modified Prim's algorithm and *k*-way partitioning). Second, to the best of our knowledge, this study is the first

Related Content

TVGuarder: A Trace-Enable Virtualization Protection Framework against Insider Threats for IaaS Environments
Li Lin, Shuang Li, Bo Li, Jing Zhanand Yong Zhao (2016). *International Journal of Grid and High Performance Computing (pp. 1-20).*
www.irma-international.org/article/tvguarder/172502

Fragment Re-Allocation Strategy Based on Hypergraph for NoSQL Database Systems
Zhikun Chen, Shuqiang Yang, Yunfei Shang, Yong Liu, Feng Wang, Lu Wangand Jingjing Fu (2016). *International Journal of Grid and High Performance Computing (pp. 1-23).*
www.irma-international.org/article/fragment-re-allocation-strategy-based-on-hypergraph-for-nosql-database-systems/165089

The Development of a Parallel Ray Launching Algorithm for Wireless Network Planning
Zhihua Lai, Nik Bessis, Guillaume De La Roche, Pierre Kuonen, Jie Zhangand Gordon J. Clapworthy (2011). *International Journal of Distributed Systems and Technologies (pp. 1-19).*
www.irma-international.org/article/development-parallel-ray-launching-algorithm/53849

Porting Applications to Grids and Clouds
Wolfgang Gentzsch (2009). *International Journal of Grid and High Performance Computing (pp. 55-77).*
www.irma-international.org/article/porting-applications-grids-clouds/2168

Adopting the Grid Computing & Semantic Web Hybrid for Global Knowledge Sharing
Mirghani Mohamed, Michael Stankoskyand Vincent Ribière (2012). *Grid and Cloud Computing: Concepts, Methodologies, Tools and Applications (pp. 1721-1736).*
www.irma-international.org/chapter/adopting-grid-computing-semantic-web/64563